



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/410,150	09/30/1999	DAVID C. BOIKE	99RSS268	6431

20594 7590 02/11/2004

CHRISTOPHER J. ROURK  
AKIN, GUMP, STRAUSS, HAUER & FELD, L.L.P.  
P O BOX 688  
DALLAS, TX 75313-0688

EXAMINER

BULLOCK JR, LEWIS ALEXANDER

ART UNIT	PAPER NUMBER
----------	--------------

2126

DATE MAILED: 02/11/2004

8

Please find below and/or attached an Office communication concerning this application or proceeding.

1

**Office Action Summary**

Application No.

09/410,150

Applicant(s)

BOIKE, DAVID C.

Examiner

Lewis A. Bullock, Jr.

Art Unit

2126

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

- 1) ☐ Responsive to communication(s) filed on \_\_\_\_.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

- 4) ☒ Claim(s) 1-39 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_ is/are withdrawn from consideration:
- 5) ☐ Claim(s) \_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 1-39 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_ are subject to restriction and/or election requirement.

**Application Papers**

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 30 September 1999 is/are: a) ☐ accepted or b) ☒ objected to by the Examiner.
- Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
- 11) ☐ The proposed drawing correction filed on \_\_\_\_ is: a) ☐ approved b) ☐ disapproved by the Examiner.
- If approved, corrected drawings are required in reply to this Office action.
- 12) ☐ The oath or declaration is objected to by the Examiner.

**Priority under 35 U.S.C. §§ 119 and 120**

- 13) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
- \* See the attached detailed Office action for a list of the certified copies not received.
- 14) ☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).
- a) ☐ The translation of the foreign language provisional application has been received.
- 15) ☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

**Attachment(s)**

- 1) ☒ Notice of References Cited (PTO-892) 4) ☐ Interview Summary (PTO-413) Paper No(s). \_\_\_\_
- 2) ☒ Notice of Draftsperson's Patent Drawing Review (PTO-948) 5) ☐ Notice of Informal Patent Application (PTO-152)
- 3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) 4. 6) ☐ Other: .

**DETAILED ACTION**

***Drawings***

1. New corrected drawings are required in this application because of Draftperson's Review. Applicant is advised to employ the services of a competent patent draftsman outside the Office, as the U.S. Patent and Trademark Office no longer prepares new drawings. The corrected drawings are required in reply to the Office action to avoid abandonment of the application. The requirement for corrected drawings will not be held in abeyance.

***Claim Rejections - 35 USC § 112***

2. Claim 25 is rejected under 35 U.S.C. 112, second paragraph, as being incomplete for omitting essential structural cooperative relationships of elements, such omission amounting to a gap between the necessary structural connections. See MPEP § 2172.01. The omitted structural cooperative relationships are: the message header to the message. See dependent claim 19 wherein it is established that the message comprises a message header. This is omitted from claim 25 that accounts for a gap between the claims.

***Claim Rejections - 35 USC § 102***

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

Art Unit: 2126

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

4. Claims 1-8, 11-18, 21-24, 26-37, and 39 are rejected under 35 U.S.C. 102(b) as being anticipated by "Microsoft Windows NT Version 4.0 Device Driver Kit" by MICROSOFT.

As to claim 1, MICROSOFT teaches a driver system interface (intermediate driver layer), comprising: an operating system (OS) interface (operating system function) to process a plurality of messages (packet / receive query / set requests) for a plurality of internal driver entities (intermediate NDIS drivers) (pg. 17, "NDIS NIC and upper layer drivers use the NDIS library to communicate with each other and to call operating system functions..."The requirements imposed on NDIS drivers dictate that they need to call, or be called by, various platform-specific operating system components...When a driver needs a buffer, it must ask the operating system to allocate memory...When a driver needs operating system support, it calls NDIS functions that, in turn can call kernal mode support routines."); and a message controller (routing functionality of NDIS intermediate driver that sends a message / request to another entity) coupled to the OS interface (operating system function) to transfer the plurality of messages (request for buffers) (pg. 17; pg. 24; pg. 34-35; pg. 37-41).

As to claim 2, MICROSOFT teaches a platform interface (via NDIS ProtocolBindAdapter function) coupled to the message controller (routing functionality of NDIS intermediate driver that sends a message / request to another entity) to provide platform specific information to the message controller (pg. 34, "SystemSpecific1 points

to a registry path if the intermediate driver stores adapter-specific information in the registry.”).

As to claim 3, MICROSOFT teaches the message controller (routing functionality of NDIS intermediate driver that sends a message / request to another entity) communicates with the OS interface through functions (invokes operating system functions) (pg. 17, “NDIS NIC and upper layer drivers use the NDIS library to communicate with each other and to call operating system functions...”The requirements imposed on NDIS drivers dictate that they need to call, or be called by, various platform-specific operating system components...When a driver needs a buffer, it must ask the operating system to allocate memory...When a driver needs operating system support, it calls NDIS functions that, in turn can call kernel mode support routines.”).

As to claim 4, MICROSOFT teaches that the intermediate driver can be layered above or below another intermediate driver and over one or more NDIS NIC drivers (pg. 24). MICROSOFT also teaches that the drivers communicate software messages (packets / receive queries / set requests) with one another through a channel (adapter binding) (pg. 34-35; pg. 37-41). Therefore, MICROSOFT teaches a plurality of message channels (adapter bindings) to communicate the plurality of messages (packet / receive query / set requests) to and from the plurality of internal driver entities (intermediate NDIS drivers) (pg. 34-35; pg. 24-25; pg. 37-41).

As to claim 5, MICROSOFT teaches the message controller (routing functionality of the initial intermediate NDIS driver) comprising: a plurality of installable components (handles to adapter bindings of lower layered drivers) corresponding to the plurality of message channels (pg. 34, "...The intermediate driver must retain this handle and...when the intermediate driver has completed its binding-related activities and is ready to accept transmit requests..").

As to claim 6, MICROSOFT teaches the plurality of installable components comprise function pointers (SystemSpecific1 / SystemSpecific2) corresponding to functions (buffer allocation) in the OS interface (pg. 34; pg. 17).

As to claim 7, MICROSOFT teaches that the intermediate driver can be layered above or below another intermediate driver and over one or more NDIS NIC drivers (pg. 24). MICROSOFT also teaches that the drivers communicate software messages (packets / receive queries / set requests) with one another through a channel (adapter binding) (pg. 34-35; pg. 37-41). Therefore, MICROSOFT teaches the message controller (routing functionality of the initial intermediate NDIS driver) routes the plurality of messages (packets / receive queries / set requests) to a plurality of internal entities (below layered intermediate NDIS drivers).

As to claim 8, MICROSOFT teaches the OS interface (operating system function) comprising: an external interface (protocol functions of the lower level intermediate NDIS driver / private interface) to communicate with the plurality of external driver entities (one or more NIC drivers / device drivers) (pg. 17, "NDIS NIC and upper layer drivers use the NDIS library to communicate with each other and to call operating system functions..."The requirements imposed on NDIS drivers dictate that they need to call, or be called by, various platform-specific operating system components...When a driver needs a buffer, it must ask the operating system to allocate memory...When a driver needs operating system support, it calls NDIS functions that, in turn can call kernel mode support routines.").

As to claims 11-13, and 17, refer to claims 1-3 and 8 for rejection. However, claim 11 further details a communications driver comprising: a network driver interface; and a miniport driver coupled to the network driver interface comprising the system interface abstraction layer and the message controller for translating the message and routing the message to an external entity. MICROSOFT teaches a communications driver (transport driver) comprising: a network driver interface (protocol functions / NDIS); and a miniport driver (intermediate NDIS driver) coupled to the network driver interface comprising the system interface abstraction layer (subsequent intermediate NDIS drivers) (pg. 24-25) and a message controller (routing functionality of NDIS intermediate driver that sends a message / request to another entity) for translating the message and routing the message to an external entity (pg. 24, "Such a driver receives

packets in a LAN format at its upper edge, translates them to another NIC-native medium format and sends them on to an NDIS miniport for that NIC...”).

As to claims 14-16, refer to claims 4-6 for rejection. However, claims 14-16 further detail each message channel for communicating a subset of the plurality of messages to and from a corresponding subset of the plurality of internal devices to a specific external device. MICROSOFT teaches that the intermediate driver can be layered above or below another intermediate driver and over one or more NDIS NIC drivers (pg. 24). MICROSOFT also teaches that the drivers communicate software messages (packets / receive queries / set requests) with one another through a channel (adapter binding) (pg. 34-35; pg. 37-41). Therefore, MICROSOFT teaches a plurality of message channels (adapter bindings) to communicate the plurality of messages (packet / receive query / set requests) to and from the plurality of internal driver entities (intermediate NDIS drivers) to an external device (NDIS NIC drivers) (pg. 34-35; pg. 24-25; pg. 37-41).

As to claim 18, MICROSOFT teaches the network driver interface further comprising: a dynamic messaging library (NDIS library) coupled to the SIAL (intermediate driver) (pg. 17, “NDIS NIC and upper layer drivers use the NDIS library to communicate with each other and to call operating system functions.”)



As to claims 21-24 and 26-29, refer to claims 11-18 for rejection. However, claim 21 further details a communications card comprising the cited driver. MICROSOFT teaches the driver architecture communicates with a communications card (pg. 12, "NDIS describes the interface by which one or more NIC drivers communicate with one or more underlying network interface cards, with one or more overlying protocol drivers, and with the operating system."). Therefore, MICROSOFT teaches the limitations as disclosed.

As to claim 30, MICROSOFT teaches a communications driver (intermediate NDIS driver), comprising: a network driver interface (NDIS) (pg. 12); and a driver system interface (functionality of a intermediate NDIS driver) comprising: an external interface (private interface) to communicate with a plurality of external driver entities (one or more NIC drivers / device drivers); and an internal interface (Miniport functions) to communicate with a plurality of internal driver entities (upper layer drivers / transport driver / other intermediate NDIS drivers) (pg. 24 – 26, especially fig. 1.1).

As to claim 31, MICROSOFT teaches the intermediate NDIS driver is layered over a NIC driver to translate between the format of the medium supported by an overlying legacy protocol driver and the format of the medium supported by an underlying NIC driver (pg. 34). For example, the intermediate NDIS driver translates packets from the overlying transports LAN format to WAN packet format and packets from the underlying NIC drivers' WAN packet format to LAN packet format (pg. 24).

Therefore, MICROSOFT teaches the external interface (protocol functions of the intermediate NDIS driver / private interface) handles the semantics of the plurality of external driver entities.

As to claim 32, MICROSOFT teaches the external interface (protocol functions of the intermediate NDIS driver / private interface) is a portion of an operating system (OS) interface (operating system function) (pg. 17, "NDIS NIC and upper layer drivers use the NDIS library to communicate with each other and to call operating system functions..."The requirements imposed on NDIS drivers dictate that they need to call, or be called by, various platform-specific operating system components...When a driver needs a buffer, it must ask the operating system to allocate memory...When a driver needs operating system support, it calls NDIS functions that, in turn can call kernel mode support routines.").

As to claim 33, MICROSOFT teaches the internal interface (Miniport functions) comprises a message controller (NDIS intermediate driver) to control a plurality of message channels (adapter bindings) to pass a plurality of messages (packet / receive query / set requests) between the plurality of external driver entities (NIC driver / device drivers) and the plurality of internal driver entities (intermediate NDIS drivers) (pg. 24; pg. 34-35; pg. 37-41).

As to claim 34, MICROSOFT teaches a method of abstracting a driver system interface, the method comprising the steps of: creating a platform specific and operating system specific message channel (adapter binding) between an internal driver entity (intermediate NDIS driver) and an external driver entity (NIC driver) (via ProtocolBindAdapter function) (pg. 34-35); and routing a software message (packet / receive query / set requests) between the internal driver entity (intermediate NDIS driver) and the external driver entity (NIC driver) through the platform specific and operating specific message channel (adapter binding) (pg. 37-41).

As to claim 35, MICROSOFT teaches that the intermediate driver can be layered above or below another intermediate driver and over one or more NDIS NIC drivers (pg. 24). MICROSOFT also teaches that the drivers communicate software messages (packets / receive queries / set requests) with one another through a channel (adapter binding) (pg. 34-35; pg. 37-41). Therefore, MICROSOFT teaches the creating of a plurality of channels between a plurality of internal driver entities (intermediate NDIS drivers) and a plurality of external driver entities (NIC drivers) and routing a plurality of software messages (packets / receive queries / set requests) between the internal and external driver entities (intermediate NDIS drivers and NIC drivers).

As to claim 36, MICROSOFT teaches the steps of: creating a platform specific and operating system specific message channel (binding) between a first internal driver entity (intermediate NDIS driver) and a second internal driver entity (another

intermediate driver) (via ProtocolBindAdapter function) (pg. 34-35; pg. 24-25); and routing a software message (packet / receive query / set requests) between the internal driver entities (intermediate drivers) through the platform specific and operating specific message channel (binding) (pg. 37-41).

As to claim 37, MICROSOFT teaches an installable components (handles to adapter bindings of lower layered drivers) corresponding to the plurality of message channels (pg. 34, "...The intermediate driver must retain this handle and...when the intermediate driver has completed its binding-related activities and is ready to accept transmit requests..").

As to claim 39, MICROSOFT teaches the internal driver entity (intermediate NDIS driver) is a miniport driver module (via storing Miniport functions in its upper edge) and the external driver entity (NIC driver) is a non-miniport driver module (pg. 34-35; and pg. 24).

### ***Claim Rejections - 35 USC § 103***

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

6. Claims 9, 10, 19, 20, 25, and 38 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Microsoft Windows NT, Version 4.0 Device Driver Kit" by MICROSOFT in view of "NDIS Concepts" by 3TECH.

As to claim 9, MICROSOFT substantially discloses the invention above. However, MICROSOFT does not mention that the message header of the message contains routing information.

3TECH teaches each message (packet) comprises a message header portion (packet header information / immediate data) (pg. 17, "An additional feature available in transmit...A protocol might use this for building packet header information...") containing routing information (source address / destination address) for the message controller and a message information portion (buffer / buffer chain in the descriptor structure of the packet) containing data related to an action for a target entity to perform (pg. 16, "These packets include all the information...including data link fields such as source and destination address."; pg. 17, "NDIS defines a structure, the transmit data buffer descriptor...and calls the MAC with the TransmitChain primitive."). Therefore, it would be obvious to combine the teachings of MICROSOFT with 3TECH in order to facilitate avoid unnecessary, time-consuming copying of data between buffers (pg. 16, last paragraph).

As to claim 10, 3TECH teaches the message header portion (packet header) comprises an event variable to indicate a unique event for a corresponding message channel (acknowledge / buffer descriptor indicating a buffer or a chain of buffers) (pg.

17, "A protocol might use this for building packet header information, or for entire small protocol-generated packets, such as acknowledgements.") and a message channel identifier (destination address) variable to indicate the corresponding message channel (pg. 16, "These packets include all the information...including data link fields such as source and destination address."; pg. 17, "NDIS defines a structure, the transmit data buffer descriptor...and calls the MAC with the TransmitChain primitive.").

As to claims 19 and 20, refer to claims 9 and 10 for rejection. However, claim independent claim 11 further details a communications driver comprising: a network driver interface; and a miniport driver coupled to the network driver interface comprising the system interface abstraction layer and the message controller for translating the message and routing the message to an external entity. MICROSOFT teaches a communications driver (transport driver) comprising: a network driver interface (protocol functions / NDIS); and a miniport driver (intermediate NDIS driver) coupled to the network driver interface comprising the system interface abstraction layer (subsequent intermediate NDIS drivers) (pg. 24-25) and a message controller (routing functionality of NDIS intermediate driver that sends a message / request to another entity) for translating the message and routing the message to an external entity (pg. 24, "Such a driver receives packets in a LAN format at its upper edge, translates them to another NIC-native medium format and sends them on to an NDIS miniport for that NIC...").

As to claim 25, refer to claim 20 for rejection.

As to claim 38, MICROSOFT substantially discloses the invention above. However, MICROSOFT does not mention that the message header of the message contains routing information.

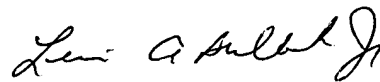
3TECH teaches each message (packet) comprises a message header portion (packet header information / immediate data) (pg. 17, "An additional feature available in transmit...A protocol might use this for building packet header information...") containing routing information (source address / destination address) for the message controller and a message information portion (buffer / buffer chain in the descriptor structure of the packet) containing data related to an action for a target entity to perform (pg. 16, "These packets include all the information...including data link fields such as source and destination address."; pg. 17, "NDIS defines a structure, the transmit data buffer descriptor...and calls the MAC with the TransmitChain primitive."). Therefore, it would be obvious to combine the teachings of MICROSOFT with 3TECH in order to facilitate avoid unnecessary, time-consuming copying of data between buffers (pg. 16, last paragraph).

### ***Conclusion***

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Lewis A. Bullock, Jr. whose telephone number is (703) 305-0439. The examiner can normally be reached on Monday-Friday, 8:30 am - 5:00 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng An can be reached on (703) 305-9678. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).



lab